**实现缓冲的镜头拉近效果**

Posted on 2013年05月14日 by U3d / [Unity3D脚本/插件](http://www.unitymanual.com/category/script)/被围观 67 次

|  |  |  |
| --- | --- | --- |
|  |  |  |

|  |  |
| --- | --- |
| 01 | **var** target : Transform; |
| 02 | **var** distance : **float** = 3.0; |
| 03 | **var** height : **float** = 1.0; |
| 04 | **var** damping : **float** = 5.0; |
| 05 | **var** smoothRotation : boolean = **true**; |
| 06 | **var** rotationDamping : **float** = 10.0; |
| 07 |  |
| 08 | **var** targetLookAtOffset : Vector3; *// allows offsetting of camera lookAt, very useful for low bumper heights* |
| 09 |  |
| 10 | **var** bumperDistanceCheck : **float** = 2.5; *// length of bumper ray* |
| 11 | **var** bumperCameraHeight : **float** = 1.0; *// adjust camera height while bumping* |
| 12 | **var** bumperRayOffset : Vector3; *// allows offset of the bumper ray from target origin* |
| 13 |  |
| 14 | function FixedUpdate() { |
| 15 |  |
| 16 | **var** wantedPosition = target.TransformPoint(0, height, -distance); |
| 17 |  |
| 18 | *// check to see if there is anything behind the target* |
| 19 | **var** hit : RaycastHit; |
| 20 | **var** back = target.transform.TransformDirection(-1 \* Vector3.forward); |
| 21 |  |
| 22 | *// cast the bumper ray out from rear and check to see if there is anything behind* |
| 23 | **if** (Physics.Raycast(target.TransformPoint(bumperRayOffset), back, hit, bumperDistanceCheck)) { |
| 24 | *// clamp wanted position to hit position* |
| 25 | wantedPosition.x = hit.point.x; |
| 26 | wantedPosition.z = hit.point.z; |
| 27 | wantedPosition.y = Mathf.Lerp(hit.point.y + bumperCameraHeight, wantedPosition.y, Time.deltaTime \* damping); |
| 28 | } |
| 29 |  |
| 30 | transform.position = Vector3.Lerp(transform.position, wantedPosition, Time.deltaTime \* damping); |
| 31 |  |
| 32 | **var** lookPosition : Vector3 = target.TransformPoint(targetLookAtOffset); |
| 33 |  |
| 34 | **if** (smoothRotation) { |
| 35 | **var** wantedRotation : Quaternion = Quaternion.LookRotation(lookPosition - transform.position, target.up); |
| 36 | transform.rotation = Quaternion.Slerp(transform.rotation, wantedRotation, Time.deltaTime \* rotationDamping); |
| 37 | } **else** { |
| 38 | transform.rotation = Quaternion.LookRotation(lookPosition - transform.position, target.up); |
| 39 | } |
| 40 | } |
| 41 |  |
| 42 |  |